**LO STUDIO DI CASO**

*Lo scopo di questo capitolo è quello di illustrare lo studio di caso utilizzato per la sperimentazione. L’applicazione realizzata è basata su uno scenario realmente esistente. Dopo aver introdotto le caratteristiche generali dello studio di caso verranno descritte le specifiche, i requisiti funzionali e di qualità, le modalità di accesso al sistema.*

**PREMESSA**

Per ottemperare agli obiettivi della tesi è stato scelto un esempio pratico di sistema e realmente esistente. E’ stato infatti individuato un dominio che potesse essere scomposto in sotto-sistemi ognuno dei quali ha un proprio insieme di requisiti, scopi e attori condividendo, tuttavia, un insieme di operazioni comuni. L’esempio applicativo è stato scelto in modo che possa essere facilmente compreso e gestito in modo da focalizzare l’attenzione sugli aspetti architetturali e tecnologici.

**CAR2GO**

**Generalità dell’applicazione**

L’applicazione di riferimento prende spunto da Car2Go, un popolare servizio di car sharing a flotta libera.

Il ***car sharing*** è un servizio che permette di utilizzare un’automobile su prenotazione, prelevandola e riportandola in un parcheggio, e pagando in ragione dell'utilizzo fatto. Questo servizio viene utilizzato all'interno di politiche di mobilità sostenibile, per favorire il passaggio dal possesso del mezzo all'uso dello stesso (cioè all'accesso al servizio di mobilità), in modo da consentire di rinunciare all'automobile privata ma non alla flessibilità delle proprie esigenze di mobilità. L'auto, in questo modo, passa dall'ambito dei beni di consumo a quello dei servizi. La novità di Car2Go è stata nell’introduzione del free floating, ossia nel fatto che non ci fossero punti di sosta convenzionati: le auto si parcheggiano per la strada, esattamente come si fa con la propria vettura. Quando un abbonato ne ha bisogno, cerca la vettura più vicina alla posizione in cui si trova, consultando il sito internet, utilizzando l’app, oppure chiamando il call center. C’è infine la possibilità di salire direttamente su un’auto parcheggiata per strada, controllando che non sia già prenotata (appare la dicitura sul parabrezza). Raggiunta l’auto designata il noleggiatore dovrà, mediante il computer di bordo della macchina, inserire alcuni dati sullo stato attuale della vettura. Tali informazioni riguardano lo stato di pulizia interna dell’abitacolo nonché la presenza di anomalie della carrozzeria della macchina. Alla fine dell’utilizzo si parcheggia, anche sulle aree di sosta riservate ai residenti. Anche nel caso di Car2Go si paga un abbonamento, e poi una cifra al chilometro comprensiva di carburante. Le auto della flotta sono tutte Smart, essendo il servizio organizzato dalla casa automobilistica. Car2Go è un servizio di respiro internazionale già attivo in oltre 25 città del mondo tra cui Roma, Milano, Firenze, Berlino, Amsterdam, Londra, Vienna, Washington, Seattle, Calgary, Vancouver.

**Lo Scenario**

L’applicazione realizzata come lavoro di tesi ha cercato di ripercorrere il più fedelmente possibile quanto avviene nella reale Car2Go. Per poter mettere in evidenza alcune peculiarità delle tecnologie utilizzate, alcuni aspetti della reale applicazione Car2Go sono stati enfatizzati mentre altri sono stati rilassati.

Un **utente**, mediante un sito web, ha la possibilità di vedere all’interno del Grande Raccordo Anulare di Roma, quali sono le vetture che può noleggiare. Di ogni vettura l’utente può osservarne la posizione sulla mappa geografica ed avere alcune indicazioni sullo stato della macchina stessa. Tali informazioni sono inerenti alla benzina attualmente disponibile nel serbatoio nonché allo stato di pulizia interno della macchina e alle condizioni esterne della carrozzeria.

Un utente, se interessato a noleggiare un’auto, può registrarsi e quindi autenticarsi al sistema. Una volta effettuato il login, un utente può decidere di effettuare la prenotazione di un’auto. In tal modo la vettura risulterà prenotata e nessun altro utente può noleggiarla. L’utente che ha effettuato la prenotazione la può disdire, rendendo l’auto nuovamente libera, oppure la può confermare decidendo cosi di iniziare un viaggio. Proprio come avviene in Car2Go, l’utente, prima di mettersi alla guida, deve compilare un form che riassume lo stato delle condizioni esterne ed interne dell’auto con eventuali note aggiuntive. Settate tali informazioni è possibile iniziare una guida. Il sistema realizzato, in realtà, effettuerà una simulazione di guida assegnando una nuova posizione dell’auto all’interno di Roma e calcolando distanza e tempi relativi allo spostamento. Tali dati sono stati ottenuti in diversi modi, quello più interessante fa uso dei WebService messi a disposizione da Google Maps per calcolare percorso e tempo di percorrenza in base alle condizioni attuali del traffico. Un utente, inoltre, può anche visionare lo storico dei noleggi effettuati ed avere un riepilogo dei costi.

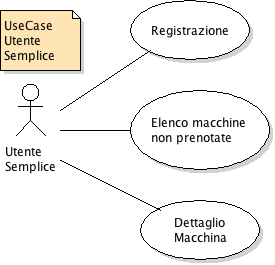
Un **amministratore** di sistema ha il duplice compito di gestire le utenze ed il parco auto. Per gestione di utenze s’indica la possibilità di assegnare il ruolo di amministratore ad un utente registrato all’applicazione. L’amministratore ha, come anticipato, il compito di dover gestire i mezzi ossia creare una nuova vettura, effettuare i rifornimenti e far pulire o riparare le auto.

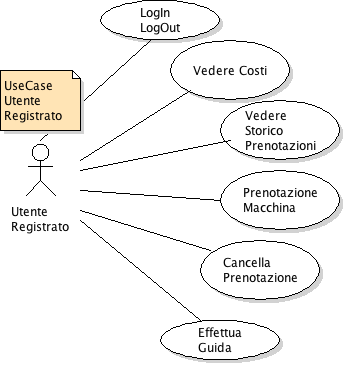
Una **vettura**, come si evince da quanto scritto precedentemente, può essere noleggiata ed ha associate delle informazioni che riguardano il suo stato interno/esterno e quantità di carburante. Di una vettura è di interesse sapere la sua posizione attuale in termine di indirizzo e coordinate quali latitudine e longitudine. Una vettura può anche inviare all’amministratore di sistema l’eventuale tipologia di guasto che l’ha colpita.

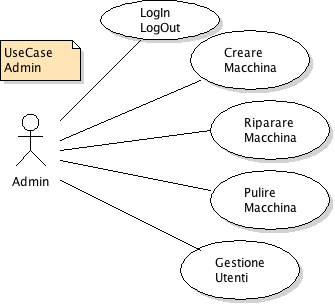
Una **prenotazione** riassume dei dati sul noleggio effettuato da un utente nei confronti di una vettura. Per tali motivi è necessario tenere traccia del quantitativo di chilometri percorsi e del tempo necessario a coprire tale distanza nonché della data in cui è stato effettuato il tragitto. Tali dati saranno necessari per effettuare la tariffazione.

**Requisiti funzionali**

Le funzionalità dell’applicazione clone di Car2Go possono essere così riassunte:







* **Gestione delle utenze**: Il sistema deve permettere la registrazione di utenze e meccanismi di autorizzazione e autenticazione. Sono consentiti tre tipi di utente: utente non registrato, utente standard registrato e amministratore. L’utente standard può interagire con il sistema noleggiando le auto e controllare lo storico dei noleggi e dei relativi costi. L’amministratore può creare auto e gestire il sistema attraverso l’invocazione di comandi per riparare, pulire e rifornire di carburante le auto. Per una migliore gestione delle utente è stato utilizzato un Roled Based Access Control
* **Gestione dei noleggi**: l’utente standard registrato al sistema può prenotare e noleggiare le auto dopo essersi autenticato al sistema. Di ogni noleggio sarà d’interesse l’ora in cui è avvenuto, la distanza percorsa e la durata dello stesso. Queste informazioni saranno utili per conteggiare i costi.
* **Gestione del parco auto**: ogni automobile della flotta ha la costante necessità di avere carburante e non problematiche interne o esterne. Per problematiche interne Car2Go intende una poca pulizia dell’abitacolo mentre le problematiche esterne riguardano la presenza di eventuali danneggiamenti della carrozzeria o eventuali guasti. Se il carburante scende sotto una soglia stabilita oppure se la vettura presenta problematiche interne/esterne sarà compito del sistema effettuare le notifiche opportune. In caso di problematica interna, la vettura comunicherà il tipo di guasto riscontrato. E’ di interesse per il sistema tenere traccia delle prenotazioni effettuate dagli utenti nei confronti di una vettura. Se una vettura è prenotata, non può essere richiesta da un altro utente.

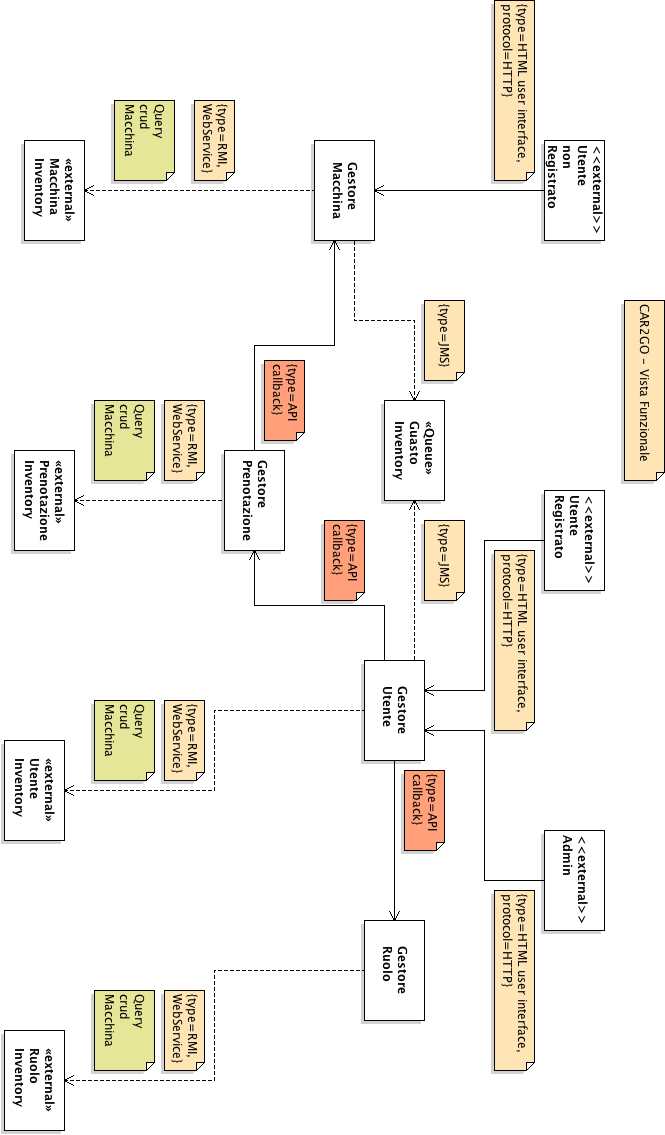
**Gestione dell’accesso ai client**

Il sistema deve essere accessibile tramite interfaccia web: a tale scopo è stata prevista la realizzazione di una web application che permetta l’accesso alle funzionalità del sistema. Tali funzionalità sono state successivamente realizzate utilizzando tecnologie diverse. Per usufruire di tali tecnologie un utente utilizza sempre le medesime pagine web. Attraverso una architettura idonea, le varie tecnologie con le quali sono state implementate le funzionalità, sono del tutto trasparenti rispetto alla logica di presentazione e quindi all’utente finale.

**Vista Funzionale**

L’immagine seguente mostra una vista funzionale dell’applicazione in cui sono mostrati:

* I tre attori esterni che interagiscono con l’applicazione:
  + Utente non registrato
  + Utente registrato
  + Admin
* Quattro componenti funzionali principali
  + Gestore Macchina
  + Gestore Utente
  + Gestore Ruolo
  + Gestore Prenotazione
* Cinque repository
  + Macchina Inventory
  + Utente Inventory
  + Ruolo Inventory
  + Prenotazione Inventory
  + Guasto Inventory
* Il tipo di comunicazione tra i componenti



**Requisiti di qualità**

Lo studio di caso selezionato permette la trattazione e l’approfondimento di problematiche relative a diversi ambiti, quali:

* **Interoperabilità**: impegno richiesto per far interagire il programma con latri programmi
* **Riusabilità**: il grado con cui un programma (od alcune sue parti) può essere utilizzato di nuovo in altre applicazioni; è correlato al modo in cui il programma è organizzato ed allo scope delle funzioni svolte dal programma;
* **Modificabilità**: facilità con cui un sistema può accomodare cambiamenti [ASW]
* **Disponibilità**: la capacità di un sistema di essere completamente o parzialmente funzionante come e quando richiesto, anche a fronte di guasti di componenti del sistema [ASW]
* **Scalabilità**: la capacità del sistema di rispondere a variazioni nel suo carico applicativo [ASW]
* **Portabilità**: la facilità con la quale il software può essere trasferito da un ambiente ad un altro come indicato dai seguenti sottoattribuiti: adattabilità, facilità di installazione, conformità e sostituibilità in particolare;
* **Testabilità**: l’impegno necessario per stabilire, tramite testing, se un programma svolge la funzione prevista.

Qualità come interoperabilità, riusabilità, modificabilità, portabilità sono state ottenute grazie un uso congiunto dello stile architetturale “Ports And Adapters (Architettura Esagonale)” di Alistar Cockburn e della metodologia Domain Driven Development. Particolare attenzione è stata quindi rivolta alla strutturazione dell’applicazione cercando inoltre di definire un modello di dominio ispirato al Model-Driven Design.